

Simplify WordPress Ch.1 – The Customizer API

BY HENNER SETYONO / ON 16 MAY 2020

#GENERAL



"Simplify WordPress" is our series of creating shortcut methods.

[Ch.1 – Customizer API](#)

[Ch.2 – Custom REST API](#)

[Ch.3 – Post Table \(coming soon\)](#)

Have you ever tried modifying the Customizer section? If yes, you definitely noticed how unnecessarily complex it is.

In this tutorial we will take a look at how to simplify it.

Terminology

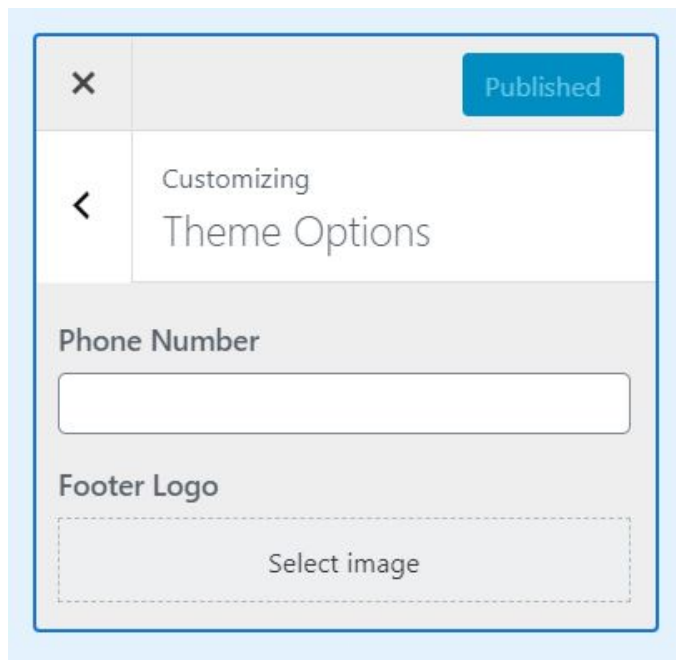


The terminology used in WP Customizer

- **SECTION** – Group of settings.
- **CONTROL** – The input method of your setting.
- **SETTING** – What's saved in the database. There are two types of setting:
 - **THEME MOD** – Bound to the theme. Changing theme will reset this.
 - **OPTION** – Bound to the site. Changing theme will not affect this.

The Original Method

Let's say we want to create a Theme Options section that contains phone number and footer's logo. Here's what it looks like and the code:



Theme Options section with 2 controls

```
add_action( 'customize_register', 'create_theme_options' );

function create_theme_options( $customize ) {
    // Create section
    $customize->add_section( 'theme_options', [
        'title' => __( 'Theme Options' ),
    ] );

    // Create setting for Phone Number
    $customize->add_setting( 'phone_no', [
        'type' => 'theme_mod',
    ] );

    // Create text control, assign 'phone_no' setting
    // and place it in 'theme_options' section
    $customize->add_control( 'phone_no_control', [
        'label' => __( 'Phone Number' ),
        'type' => 'text',
        'section' => 'theme_options',
        'settings' => 'phone_no',
    ] );

    // Footer logo
    $customize->add_setting( 'footer_logo', [
        'type' => 'theme_mod',
    ] );

    $customize->add_control(
        new WP_Customize_Image_Control(
            $customize,
            'footer_logo_control',
            [
                'label' => __( 'Footer Logo' ),
```

```

        'section' => 'theme_options',
        'settings' => 'footer_logo',
    ]
    )
);
} // close function

```

PHP

In my opinion, those methods are too wordy and repetitive. Having to define the Setting and Control method separately is also questionable.

Our Simplified Methods

We have created a helper class that mimics the functionality of the original:

Get the Helper Class code here

```

require_once __DIR__ . '/my-customizer.php';
add_action( 'customize_register', 'create_theme_options' );

function create_theme_options( $customize ) {
    $customize = new MyCustomizer( $customize );

    // No need to define title, it will be derived from the slug
    $customize->add_section( 'theme_options' );

    // Setting and Control method are combined
    $customize->add_setting( 'phone_no', [
        'type' => 'theme_mod',
        'control' => [
            'type' => 'text',
            'section' => 'theme_options'
        ]
    ] );

    // No longer need to pass in a Class object to define Image control
    $customize->add_setting( 'footer_logo', [
        'type' => 'theme_mod',
        'control' => [
            'type' => 'image',
            'section' => 'theme_options'
        ]
    ] );
}

```

PHP

Available input types:

- `text`
- `select` – require `choices` argument that contains `values => labels` array.

- `radio` – also require `choices` argument.
- `checkbox` – just a single on/off checkbox.
- `textarea`
- `dropdown-pages`
- `email`
- `url`
- `number`
- `hidden`
- `date`
- `image`
- `cropped_image` – Image that allows cropping. [Detail here »](#)
- `color`
- `upload`
- `visual_editor` – textarea with bold, italic, etc.
- `code_editor` – require `code_language` argument with possible value: clike, css, diff, htmlmixed, http, javascript, jsx, markdown, gfm, nginx, php, sass, shell, sql, xml, and yaml

• • •

Conclusion

Customizer is an underused feature in WordPress. There are many reasons, but being harder to modify than a traditional Setting page is one of them.

Hopefully with this helper class, you can give Customizer a try and see how useful it is.

In Chapter 2, we will take a look at the dreaded Post Admin Table API.

Let me know in the comment below if you have feedback or question regarding Customizer API 😊

Useful Links:

- [Section arguments list](#)
- [Setting arguments list](#)
- [Control arguments list](#)

- [MyCustomizer helper class](#)

php

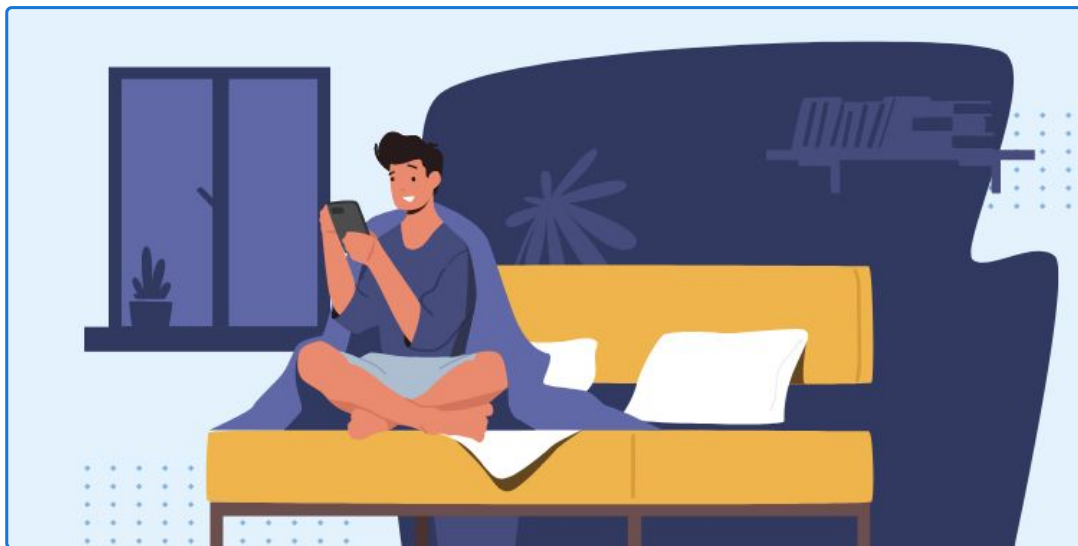


Written by

Henner Setyono

A web developer who mainly build custom theme for WordPress since 2013. Young enough to never had to deal with using Table as layout.

Related Posts



Quick and Easy Dark Mode without Plugin

On 2 Aug 2022



Our WordPress Workflow with Github Action

On 28 Feb 2022 / 2 Comments



What is WordPress Nonce? (Preventing CSRF Attack)







On 14 Aug 2020

Leave a Reply

Name *

Email *

Website

B **I**      Block 

- ☐ Save my name, email, and website in this browser for the next time I comment.
- ☐ Notify me of follow-up comments by email.

Post Comment



.dev

Design by [Pixel Studio](#)

About wpTips.dev

wpTips is a non-profit blog focused on **advanced** WordPress tutorial in a concise and beautiful way.

We want to show the world the full potential of our beloved CMS.

Join Our Community

We are a team of enthusiasts who enjoy sharing our knowledge.

Let us know at hello@wptips.dev if you are interested to write some tips!

Copyright © 2022 WPTips.dev